In the Claims:

Please cancel Claims 1-6, 11, 13 and 17; amend Claims 7 and 12; and add new Claims 18-29, all as shown below. Applicant respectfully reserves the right to prosecute any originally presented or canceled claims in a continuing or future application.

1-6. (Canceled).

7. (Currently Amended) A system for memory management comprising:

a computer system including a virtual machine operating thereon;

a memory space within said computer system and accessible by the virtual machine for the runtime storage and execution of applications; and,

a garbage collector that uses a reinforcement learning process to control the allocation of memory to applications within ~~said~~ the memory space;

wherein the garbage collector performs the steps of

measuring system-wide and application-specific parameters to determine a current state of the memory space,

monitoring system events that contribute to the state,

performing actions by the garbage collector to adjust the allocation of memory,

calculating a reward, and measuring the system-wide and application-specific parameters to determine a new state associated with the reward, and,

repeating the steps as necessary to control the allocation of memory to applications within the memory space.

8. (Original) The system of claim 7 wherein the virtual machine is a Java Virtual Machine.

9. (Original) The system of claim 7 wherein the reinforcement learning uses a temporal difference method.

10. (Original) The system of claim 9 wherein the temporal difference method uses on-line SARSA.

11. (Canceled).

12. (Currently Amended) A method for memory management comprising the steps of:

analyzing the memory ~~or storage~~ space of a computer system or virtual machine; and,

using a garbage collector together with a reinforcement learning ~~technique~~ process to control the ~~management~~ allocation of memory to applications within ~~of~~ the memory ~~or storage~~ space

wherein the garbage collector performs the steps of

measuring system-wide and application-specific parameters to determine a current state of the memory space,

monitoring system events that contribute to the state,

performing actions by the garbage collector to adjust the allocation of memory,

calculating a reward, and measuring the system-wide and application-specific parameters to determine a new state associated with the reward, and,

repeating the steps as necessary to control the allocation of memory to applications within the memory space.

13. (Canceled).

14. (Original) The method of claim 12 wherein the virtual machine is a Java Virtual Machine.

15. (Original) The method of claim 12 wherein the reinforcement learning uses a temporal difference method.

16. (Original) The method of claim 15 wherein the temporal difference method uses on-line SARSA.

17. (Canceled).

18. (New) The system of claim 7, wherein the system-wide and application-specific parameters to determine a current state include the amount of allocated memory per time unit.

19. (New) The system of claim 7, wherein the system-wide and application-specific parameters to determine a current state include the amount of allocated memory the last time a decision to garbage collect was made.

20. (New) The system of claim 7, wherein the system-wide and application-specific parameters to determine a current state include how much of the memory space is fragmented.

21. (New) The system of claim 7, wherein the system-wide and application-specific parameters to determine a current state include any of the average size of new allocated objects, average age of allocated objects, or average amount of new allocated objects.

22. (New) The system of claim 7, wherein the system events that contribute to the state include any of a variable representing if a garbage collection was made during the last time step, a variable representing if the system ran out of memory during the last time step, the amount of occupied memory before the garbage collection, the amount of occupied memory left after completed garbage collection, or the number of situations where a heap lock needed to be taken.

23. (New) The system of claim 7, wherein the actions by the garbage collector to adjust the allocation of memory include whether to garbage collect or not, whether to extend or compact the memory space, and by how much.

24. (New) The method of claim 12, wherein the system-wide and application-specific parameters to determine a current state include the amount of allocated memory per time unit.

25. (New) The method of claim 12, wherein the system-wide and application-specific parameters to determine a current state include the amount of allocated memory the last time a decision to garbage collect was made.

26. (New) The method of claim 12, wherein the system-wide and application-specific parameters to determine a current state include how much of the memory space is fragmented.

27. (New) The method of claim 12, wherein the system-wide and application-specific parameters to determine a current state include any of the average size of new allocated objects, average age of allocated objects, or average amount of new allocated objects.

28. (New) The method of claim 12, wherein the system events that contribute to the state include any of a variable representing if a garbage collection was made during the last time step, a variable representing if the system ran out of memory during the last time step, the amount of occupied memory before the garbage collection, the amount of occupied memory left after completed garbage collection, or the number of situations where a heap lock needed to be taken.

29. (New) The method of claim 12, wherein the actions by the garbage collector to adjust the allocation of memory include whether to garbage collect or not, whether to extend or compact the memory space, and by how much.